DEPLOYING QoS IN SERVICE PROVIDER NETWORK

BACKBONE QoS FOR LAYER3 VPN

Andy Chien

Consulting System Engineer

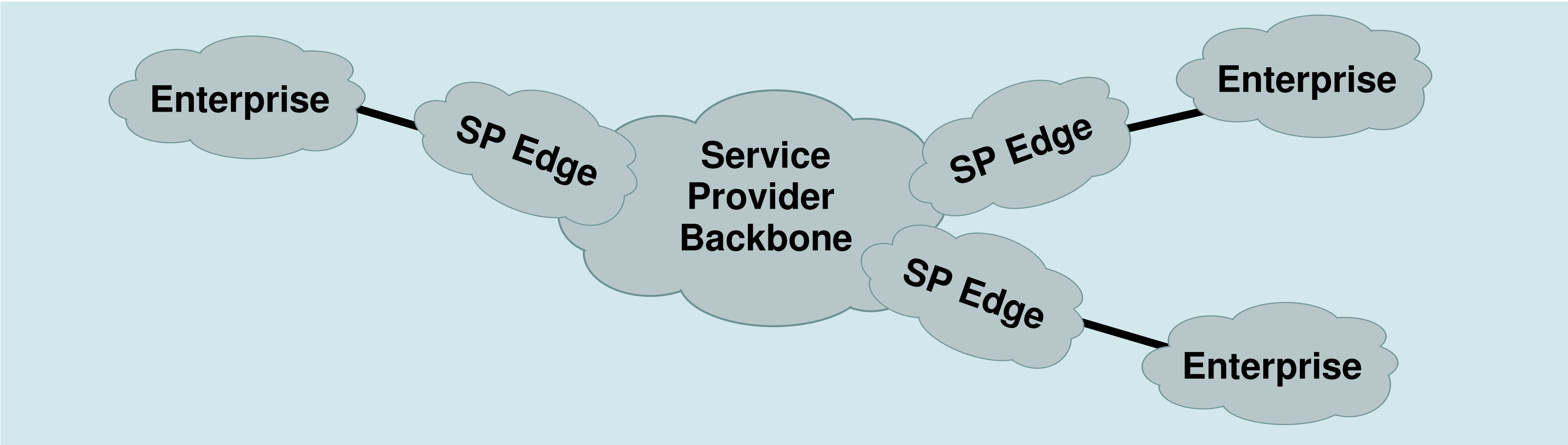
Cisco Systems

hchien@cisco.com

Backbone QoS Options

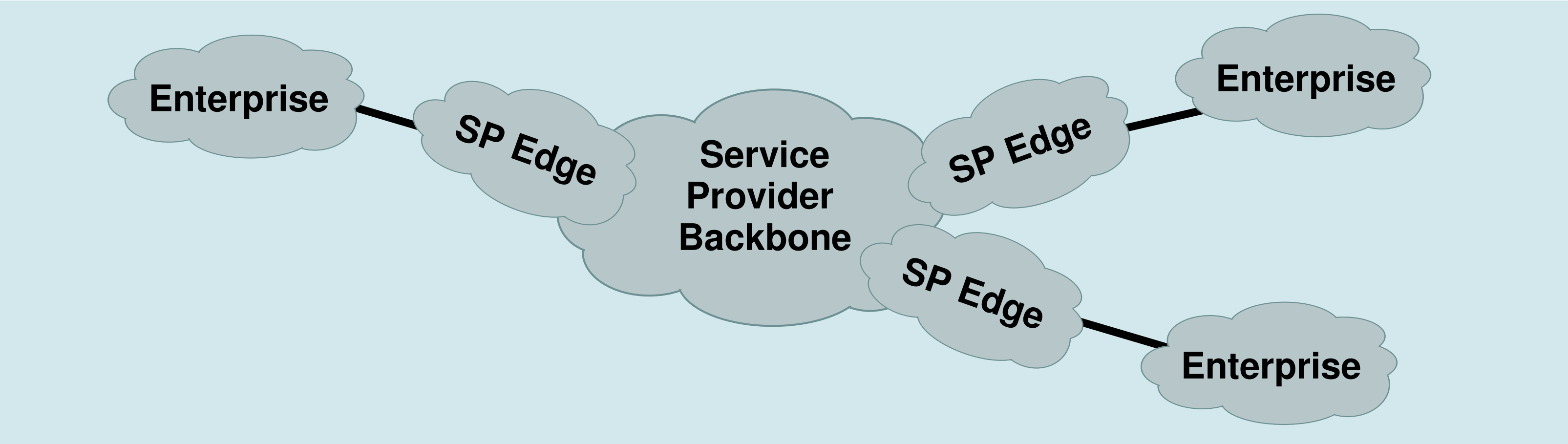
2

Backbone QoS Options



- **Overprovisioned backbone**
- **DiffServ**
- **Traffic Engineering**
- **DiffServ Aware Traffic Engineering**

Backbone QoS Options

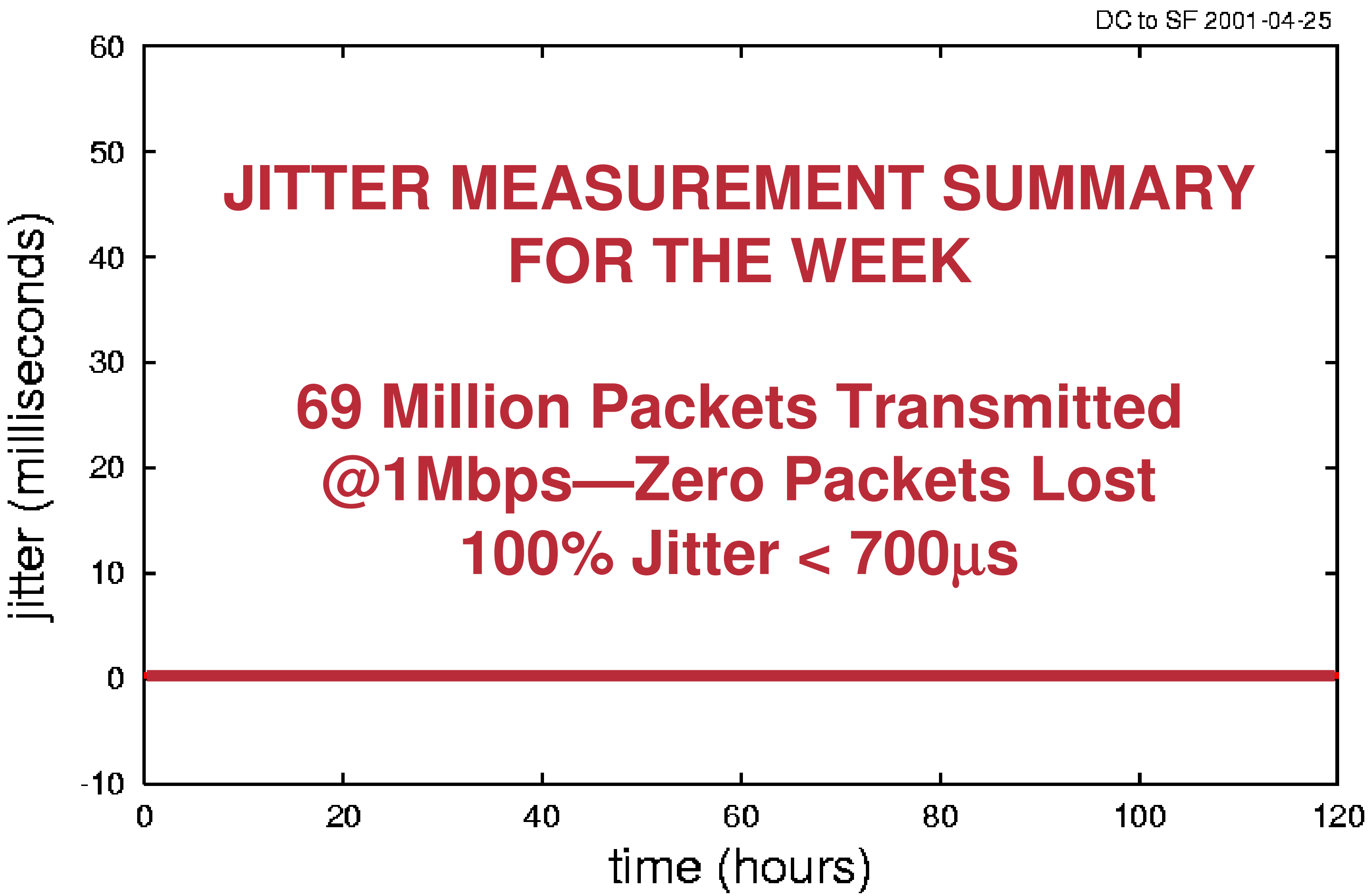


- **How to design an IP backbone for low delay, jitter, and loss? Or, is Diffserv really needed in the backbone?**
- **A simple solution is:**

  Over provision by ~2x the max. aggregate traffic load [CASNER]

  Over Provisioning Factor (OP Factor) = available BW/load

Overprovisioning Option



Source: Stephen Casner, NANOG 22

Drawbacks of Overprovisioning

- Risk related to provisioning failure
- Fate sharing!

  No isolation between VPN, VoIP, Internet
- Expensive

  Design for the aggregate!

Provisioning Failure

- **Capacity planning failures**
  - **Small overprovisioning ratio**
- **Unexpected traffic demands**
- **Network failure situations**
- **Internet DoS attack**

Such provisioning failure affects voice traffic

DiffServ: Benefits

1. **Traffic Class Isolation**:
   - **DiffServ used just for class Isolation**
     - Isolates high priority traffic classes from the effect of provisioning failure
2. **Bandwidth Efficiency**:
   - **Efficiency via over or under-provisioning of classes**
     - Without DiffServ, bandwidth for each class is over-provisioned with the same OP-factor as VoIP - not very efficient
     - Diffserv in backbone allows multiple classes of traffic with different under- or over-provisioning ratios per class – hence bandwidth efficiency can be achieved by using smaller OP-factors for non-VoIP classes

e.g., OP-factors 2 (voice), 1.2 (business) , and 1 (best effort)

Efficiency Gain with DiffServ

- Greater potential efficiency gain, if the traffic requiring the highest SLA targets (requiring the maximum OP-ratio) is a small proportion of the total traffic
- Efficiency Gain can be realized as:

  EITHER- less bandwidth requirement (vs. non-DiffServ case) to achieve the same SLA

  OR - more aggregate traffic support (vs. the non-Diffserv case) for the same provisioned bandwidth

Backbone SLAs and High-Level DiffServ Design

© 2005 Cisco Systems, Inc. All rights reserved.

10

Backbone Classes

Class	Throughput	Availability	Sequence	Latency	Jitter	Loss
Realtime	Y	Y	Y	Y	Y	Y
Business	Y	Y	Y	Y		Y
Best Effort	Y	Y	Y			Y

- **Three aggregate classes in the core**
- **More classes at the edge**
  - Realtime: VoIP, Video
  - Business: lat-optimized, th-optimized
- **RP and management traffic treated as Bus class in backbone**
  - Bus. class engineered for low delay and low loss
  - Reduces number of classes supported
  - Simplifies design, configuration and capacity planning

SLAs: Realtime Class: Delay/Jitter

- If X = max backbone queuing/switching delay (per end-to-end delay budget), then
  - Per-hop delay budget = X/N, where N = max number of hops
- Example: If queuing/switching delay budget for core = 5 ms and max # of hops = 10, then
- Per-hop delay = 5ms/10 = 500 us

Note: Service Providers May Choose Different Per-Hop Delays Depending on Their Network

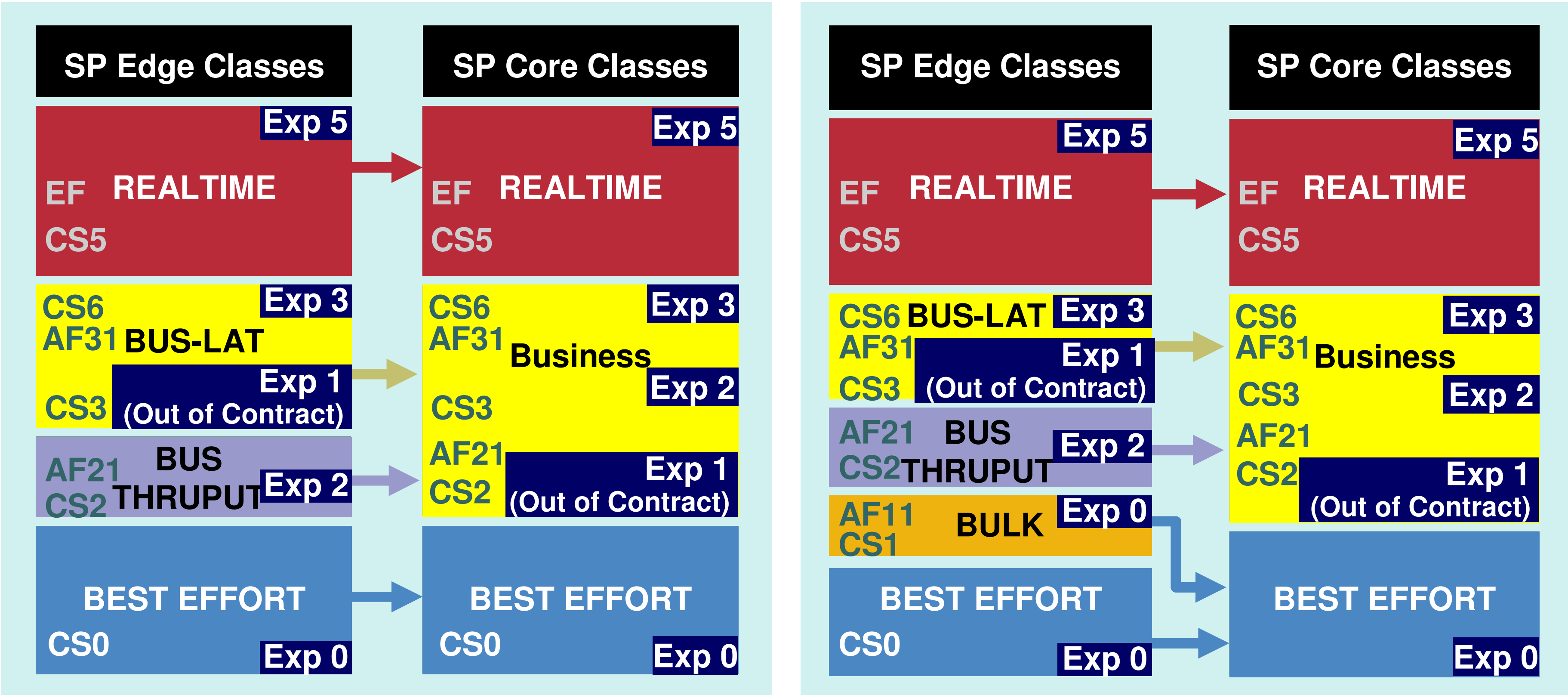
Proportional Differentiation Model

- **Backbone is dimensioned to avoid congestion for the majority of the time**
- **If congestion occurs, BE traffic may get dropped while Realtime and Business classes are dimensioned to be preserved**
- **Tighter SLA for VoIP than for Business, and for Business class than BE**

Backbone Low Level DiffServ Design

Backbone Traffic Classes and Markings

Examples of Edge Class to Core Class Mapping



Additional SP Specific Classes, i.e., Routing/NMS, if any, Are Combined with ‘Business Class”

Marking Scheme

- **SP may choose to use CS0–CS6 rather than EF, AFxy**
  - **Eases translation from/to MPLS-EXP, 802.1P, and 802.17 (resilient packet ring)**
  - **Allows backward compatibility with systems which only support IP Precedence**
- **Default action of copying IP precedence to MPLS EXP may be used, if desired if they match (generally not suitable for QoS Policies)**

Realtime Class: PHB/Scheduling

- **VoIP class ➔ EF PHB/PQ**
- **To ensure delay/jitter SLA is met, it is accepted that Realtime traffic should be < x% of physical link speed**

  [BONALD], [CASNER], [CHARNY]
- **But what is x?**

  From [CHARNY]—worst-case analysis suggests that x could be as low as 15%

  From [BONALD]—statistical and simulation results suggest that x could be as high as 75%

  From [CASNER]—real-life measurements suggest that x could be above 50%

Realtime Class: OP Factor

- A Service provider's OP Factor target may vary
- Example:
  - Assume a worst case realtime traffic of < 50% of link speed
    - Hence x < 25% of physical link speed in non-failure conditions
      - Assuming worst case traffic is double the normal load
      - Traffic matrix + "what-if" scenarios will provide more accurate target
  - Hence very unlikely that PQ will starve other queues
    - Strict PQ implementation optimal for delay/jitter

VoIP: Policer

- **Capacity planning should ensure EF load < 0.25**
- **No need to police… except if**

  Fear DoS attack into EF

  Fear burst aggregation and you prefer lost packets over jittered packets

- **If Policing is used, then**

  Policer's rate = your target max EF rate per hop under failure (eg. 50%)

  Policer's burst = your target max jitter per hop * linerate

Business Class: PHB/Scheduling

- **AF PHB**
- **Over Provision Business Class bandwidth so its packets can be drained faster ➔ low queue size ➔ lower latency**
- **Best Effort Class is not overprovisioned to this extent**
- **Example:**
  - Allocate 90% of remaining bandwidth (once PQ has been serviced) to Business Class
  - 90% of (100–25) = 67% (no failure, min VoIP)
  - 90% of (100–50) = 45% (failure, max VoIP)
  - Note: In this example, SLA should take in it to account that business traffic could only be 45% of link bandwidth during congestion
- **Expected load < 67%**
- **WRED configured to:**
  - Implement out-of-contract dropping policy
  - Optimize TCP throughput

Best Effort: PHB/Scheduling

- **AF PHB**
- **Allocate the rest of the bandwidth**
- **Example:**

  10% of remaining bandwidth once PQ has been serviced

  10% of (100–25) = 7% (no failure, min VoIP, max Bus)

  10% of (100–50) = 5% (failure, max VoIP, max Bus)

  Expected load >> 7%, but OK, since it uses available bandwidth from other classes

- **WRED to optimize TCP throughput**

MQC Configuration

```
class-map match-any RT
  match ip dscp 40
  match mpls experimental 5
class-map match-any BUS
  match ip dscp 24 32 48 8
  match mpls experimental 4 6 3 1
!
policy-map oc48_policy
  class RT
    priority
    police 1200000000 150000000
      conform transmit exceed drop
  class BUS
    bandwidth remaining percent 90
    random-detect
    random-detect pre 3 1500 9692 1
    random-detect pre 4 1500 9692 1
    random-detect pre 6 1500 9692 1
    random-detect pre 1 500 1012 1
  class class-default
    bandwidth remaining percent 10
    random-detect
    random-detect pre 0 1500 9692 1
```

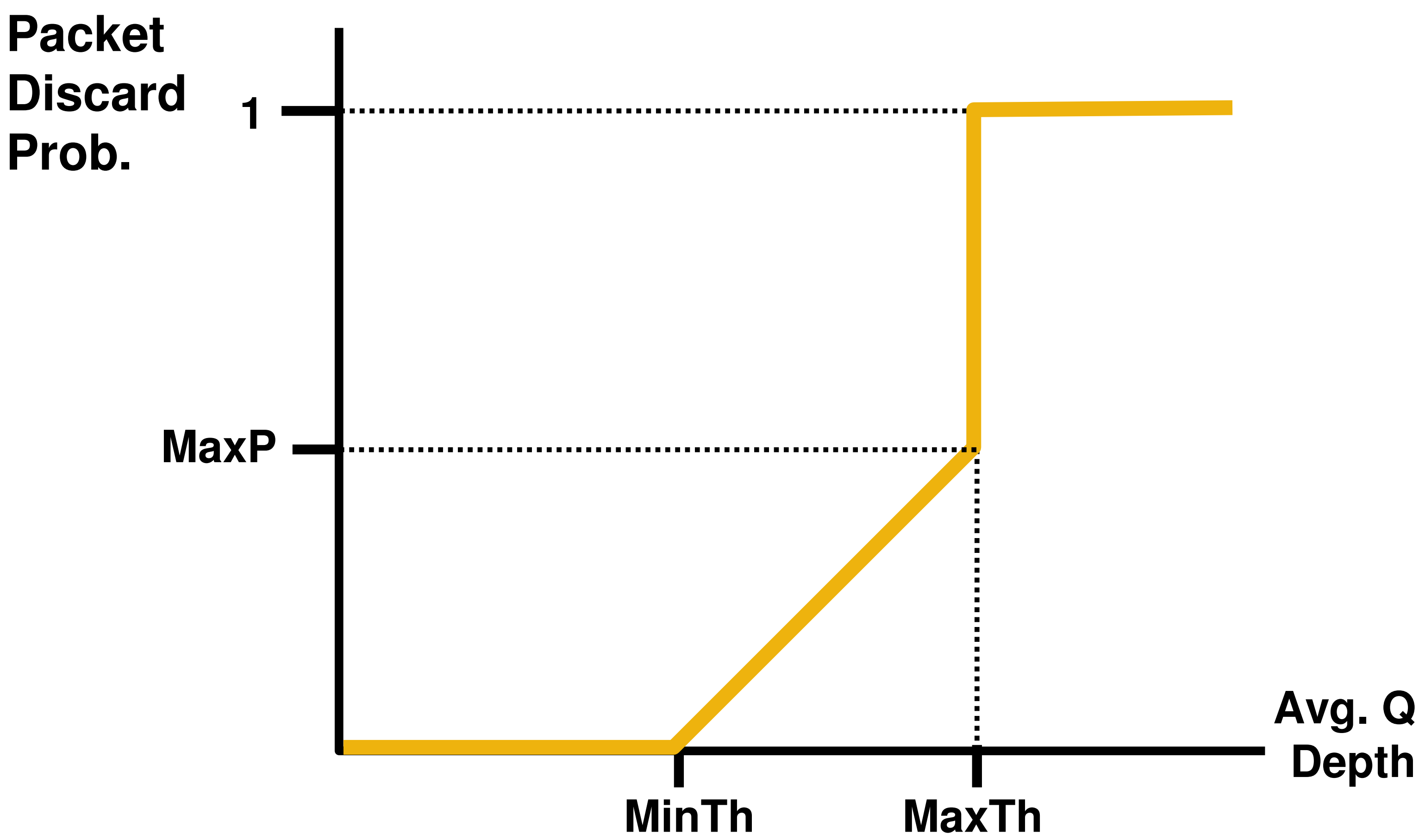
Guidelines on RED Tuning

- **WRED tuning is a complex problem that depends upon many factors, including:**
  - **The offered traffic load and profile**
  - **The ratio of load to available capacity**
  - **The behaviour of traffic in the presence of congestion**
- **These factors vary network by network**
  - **Dependent upon services offered and on customers using those services**

Guidelines on RED Tuning

- **As a starting point, following slides give some generic guidelines**
  - **Suggested for ➔ 10Mbps**
  - **Recommend fine-tuning based upon testing and operational experience in each specific environment**
- **The goal is to maximize the link utilization while minimizing the mean queue depth (hence delay)**

Guidelines on RED Tuning



Guidelines on RED Tuning

- **Min threshold value should be high enough to maximize the link utilization**

  If too low, packets may be dropped unnecessarily, and the link will not be fully utilized

- **Min threshold = 0.15 * P**

  Where P is the pipesize = RTT * BW/(MTU * 8)

  Use 1500 byte packets for the MTU even if MTU is configured at 4470

  For OC48 rate, P can be calculated as:

  P = 100ms * 2.5Gbps/(1500 * 8) = 20,000

Guidelines on RED Tuning

- **Difference between the max threshold and the min threshold should be large enough to avoid global synchronization**

  If difference is too small, many packets may be dropped at once, resulting in global synchronization

- **Max threshold = 1 * P**

  Where P is the pipe size = RTT * BW/(MTU * 8)

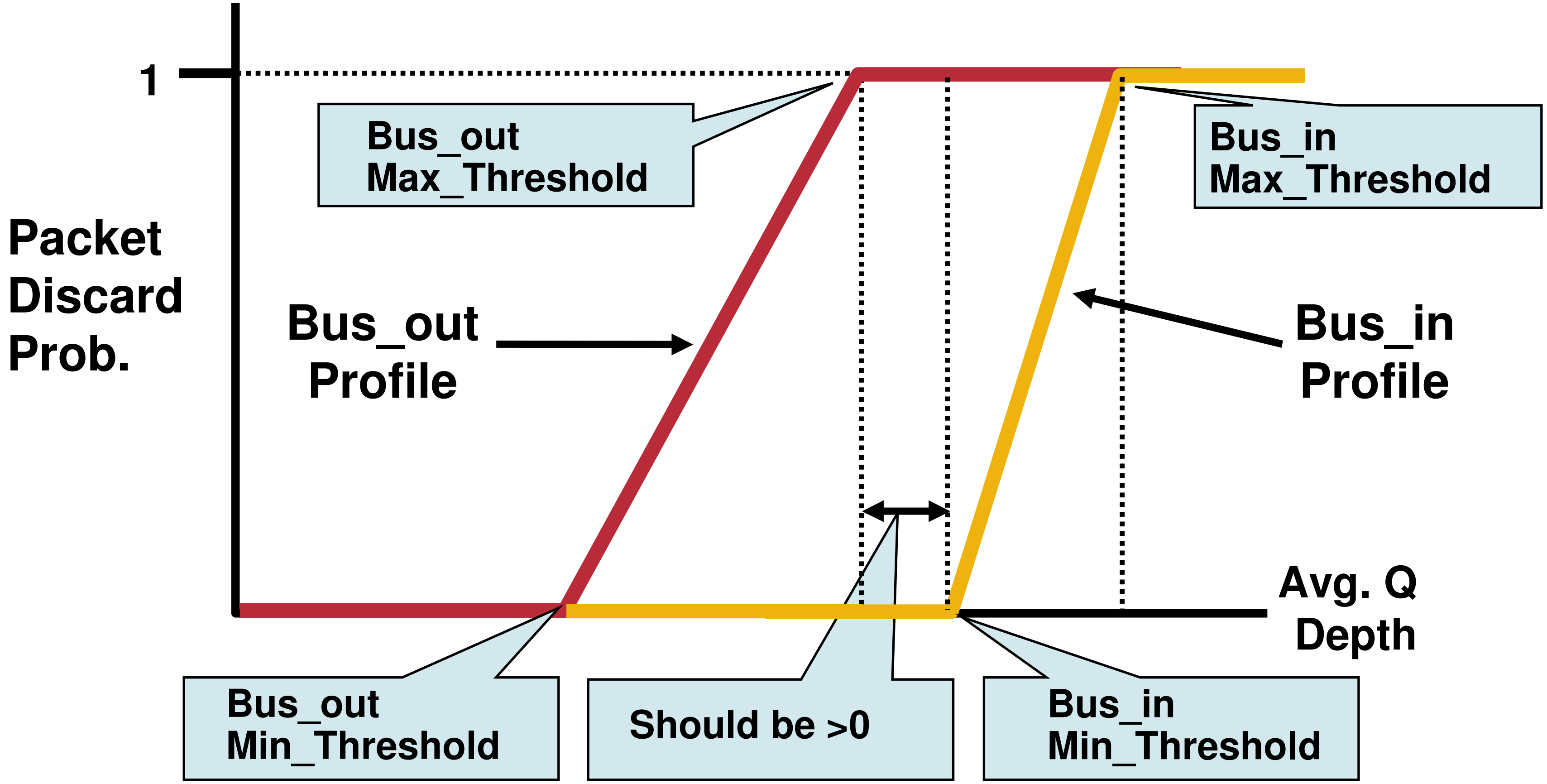
- **Set MaxP = 1**

Guidelines on RED Tuning

Link Speed	P	MinTh	MaxTh
OC3/STM-1	1292	97	609
OC12/STM-4	5184	389	2437
OC48/STM-16	20000	1500	9692

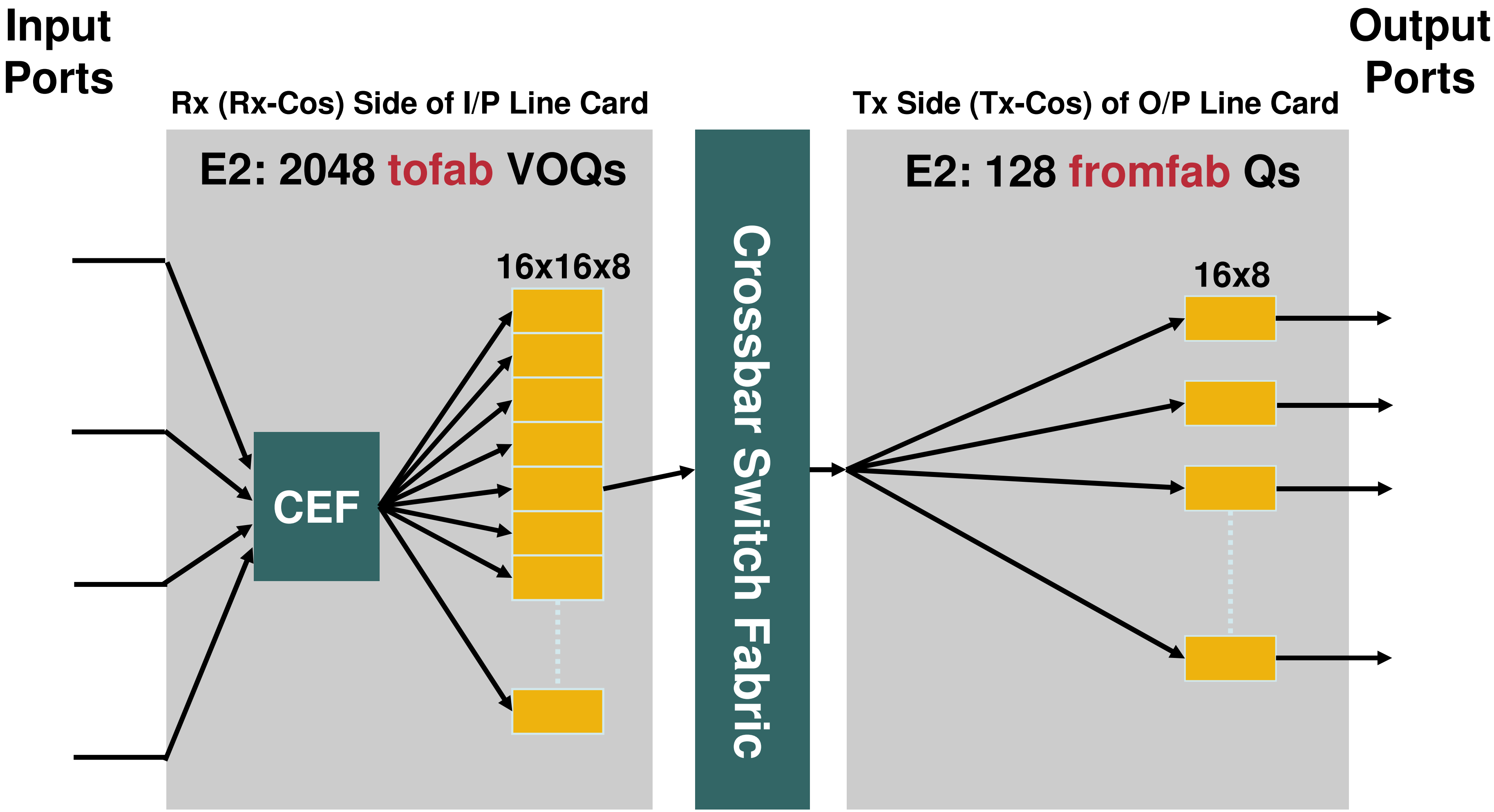
- **Based on simulations, these recommendations ensure at least 85% utilization with a mean queue size below 0.2P**
  - **Thus a mean queuing delay less than 20ms**
- **Expected per class load**
  - **Assumed 50% BUS, 50 % BE**
- **Note: On GSR, maxTh is adjusted such that (maxTh–minTh) is a power of 2**

RED Tuning: Bus_in/Bus_out



Min_threshold of in_contract > Max_threshold of Out-of-Contract Traffic

Configuring to-Fab Queuing



When Is to-Fab QoS Needed?

- GSR is similar to a network of routers (linecards) interconnected by trunks (crossbar fabric)
- Bandwidth of trunks >> than egress bandwidth of the router, and hence stable congestion is unlikely
- Short-term congestion due to very high average load and burstiness aggregation is thus rare but is possible
- Recommendation: configure to-fab queuing to provide SLA assurance

Configuring to-Fab Queuing

- **Bandwidth allocation most important**

Realtime	➔	PQ
Business	➔	90% of remaining bandwidth
BE	➔	10% of remaining bandwidth

- **WRED tuning is much less important on to-fab**

  If congestion arises there it will be due to burst aggregation… which is actually not the focus of RED

  WRED filters out short bursts and only reacts to steady congestion

To-Fab Configuration:

```
slot-table-cos SLOT_TABLE
 destination-slot all oc48_policy
!
rx-cos-slot all SLOT_TABLE
!
cos-queue-group oc48_policy
  precedence 0 queue 0
  precedence 1 queue 0
  precedence 3 queue 1
  precedence 4 queue 1
  precedence 6 queue 1
  precedence 5 queue low-latency
  precedence 0 random-detect-label 1
  precedence 1 random-detect-label 0
  precedence 3 random-detect-label 1
  precedence 4 random-detect-label 1
  precedence 6 random-detect-label 1
  random-detect-label 0 500 1012 1
  random-detect-label 1 1500 9692 1
  queue 0 1
  queue 1 71
  queue low-latency strict
```

To-Fab Configuration Only Currently Supported with Legacy GSR CLI

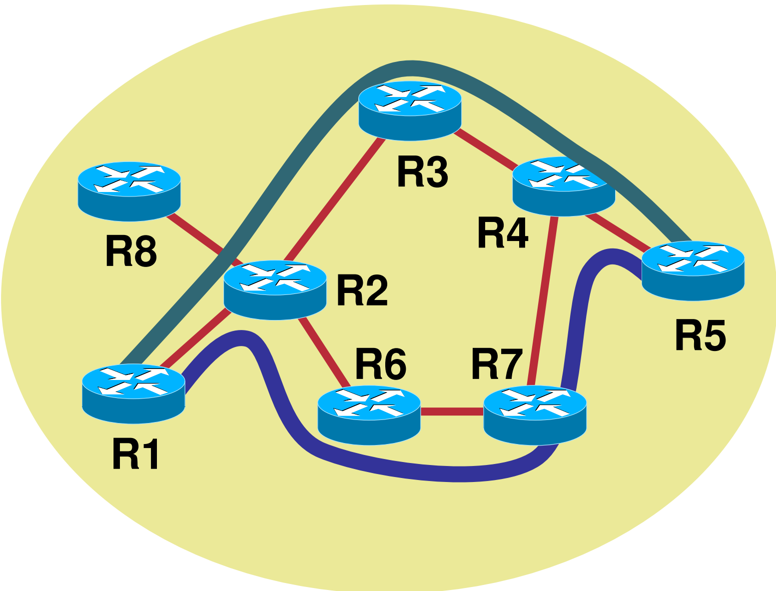
QoS with Traffic Engineering

34

MPLS Traffic Engineering and QoS

Network Engineering: Designing Network per Expected Traffic
Traffic Engineering (TE): Fitting Traffic to an Existing Network
MPLS Traffic Engineering (MPLS TE) Is TE Using MPLS

Detailed Treatment of MPLS TE—in RST 2603

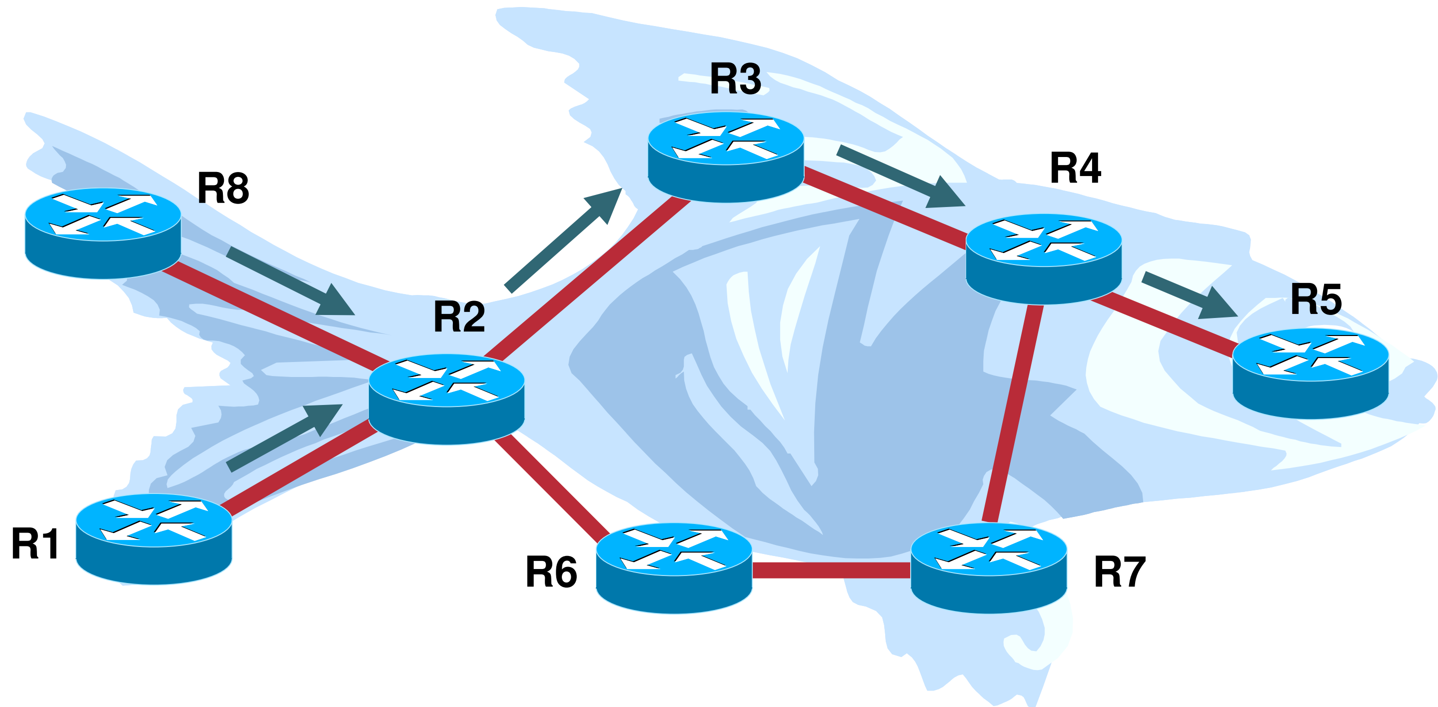


Enables Multiple Tunnels Between Two Routers

Tunnel Traffic Differentiation Based on:

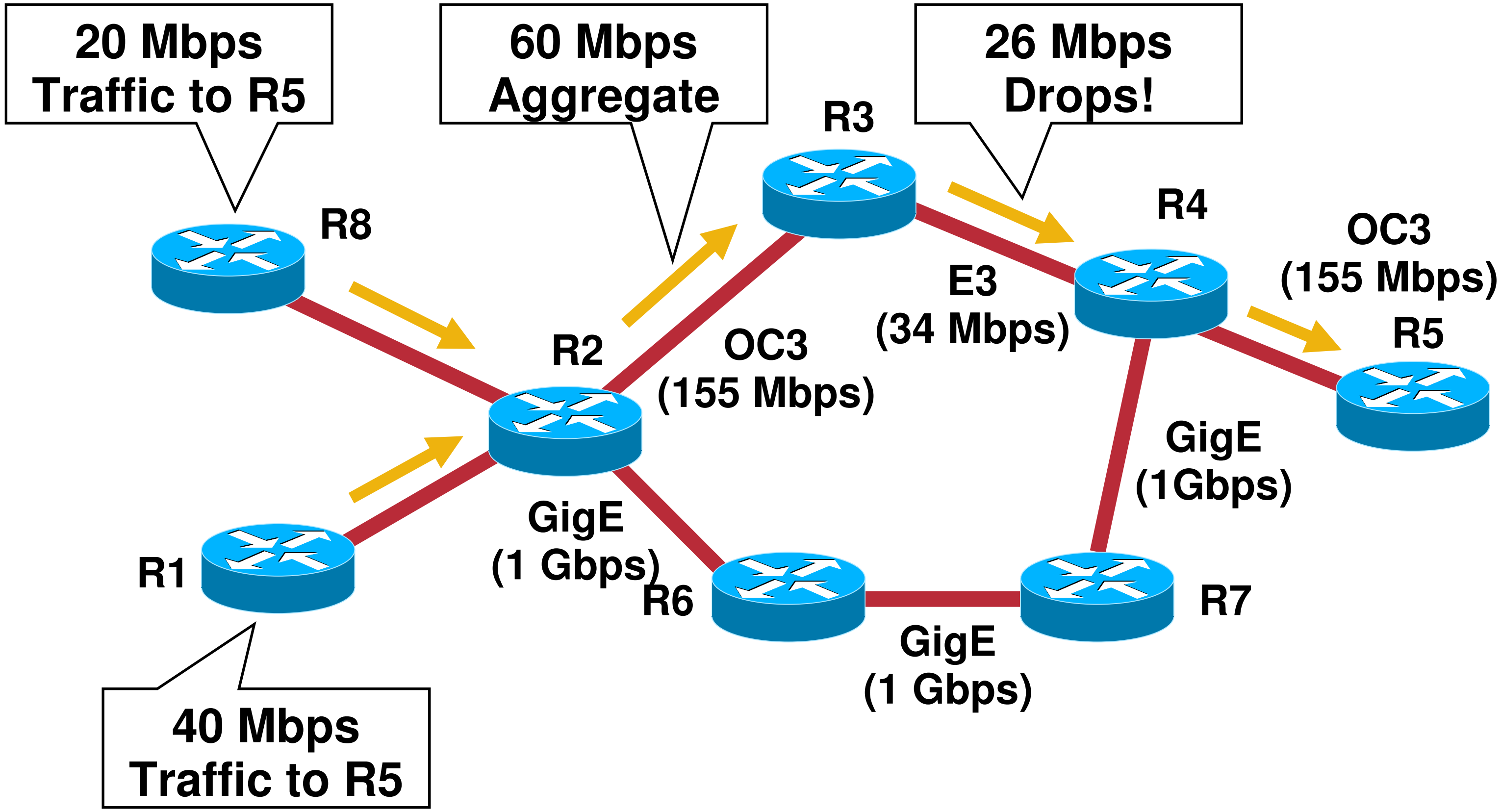
- Customer network/interface
- VRF
- Traffic class (DSCP, EXP)

MPLS TE and QoS
Basics of Traffic Engineering: The Fish Problem

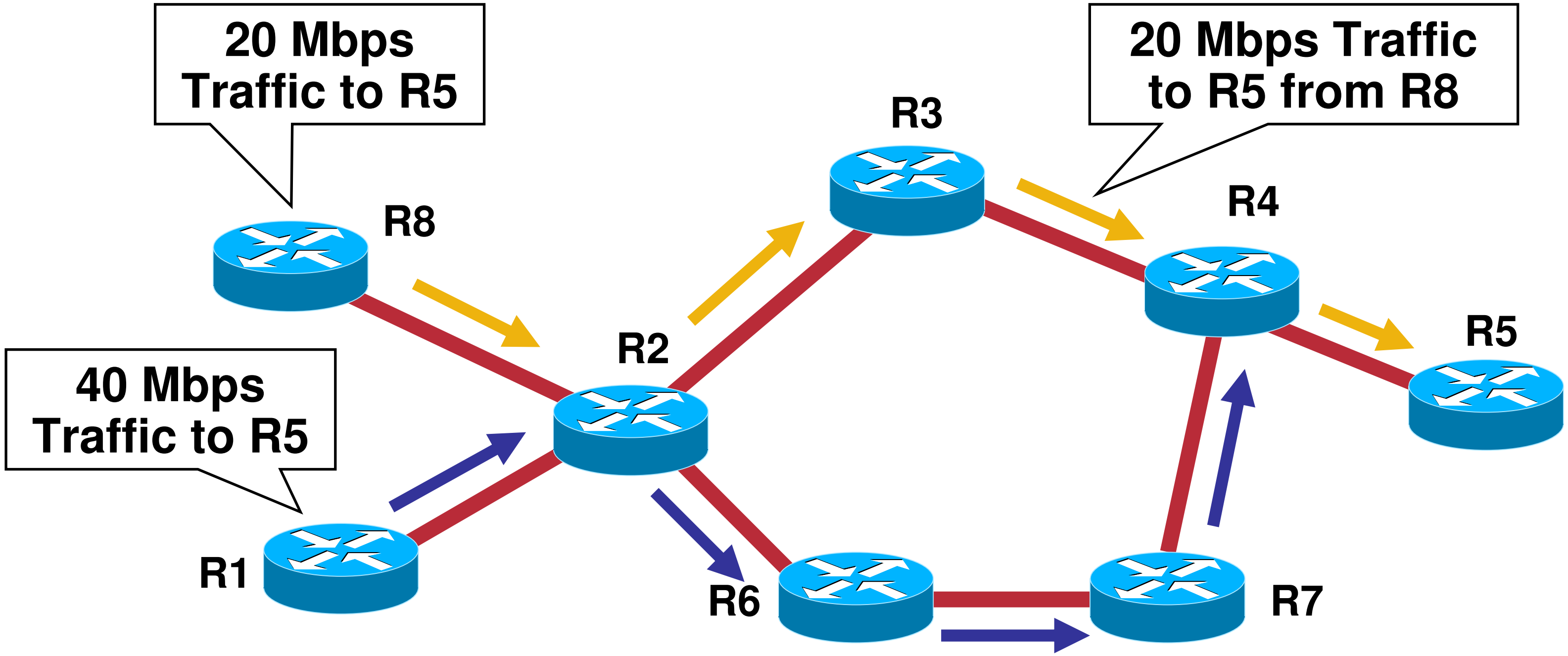


- **IP shortest path destination-based routing may congest the shortest path, while alternate paths, if any, are underutilized**
- **TE enables traffic distribution through multiple paths (tunnels) to alleviate this**

Traffic Engineering Basics: The Fish Problem



Traffic Engineering Basics: Fish Problem and TE



- MPLS labels can be used to engineer explicit paths
- Tunnels are UNIDIRECTIONAL

Normal path: R8 → R2 → R3 → R4 → R5

Tunnel path: R1 → R2 → R6 → R7 → R4

Traffic Engineering Basics

- **Explicit routing—a tunnel path can have a user specified series of next hops (like static routing)**
- **Constraint-based routing and admission control—**

    Automatic/explicit tunnel path selection based on bandwidth required by the tunnel, and bandwidth availability on each hop along the path

    Tunnel not created unless bandwidth is available (admission control)

    Bandwidth allocation in Control Plane only

    RSVP-TE advertises link attributes

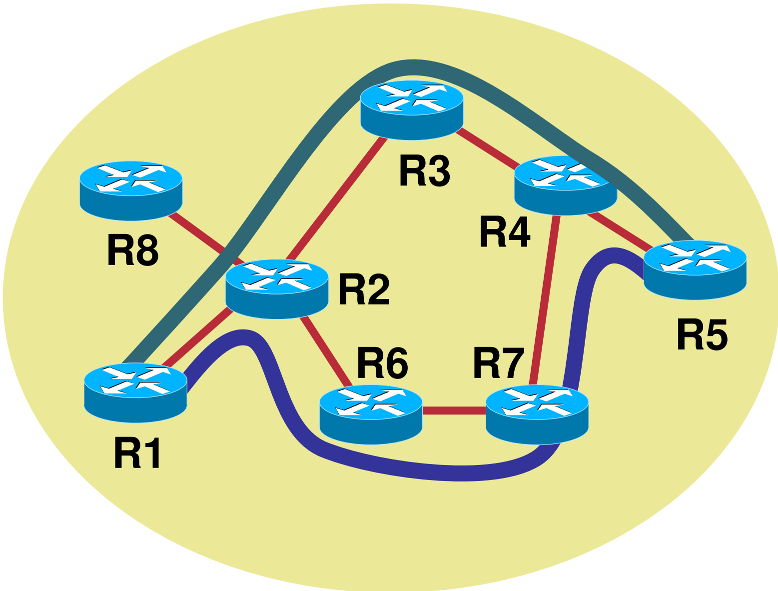
    ISIS and OSPF extensions

- **Protection**

    Fast Route Recovery (FRR) to switch packets after link, node, or path failures (node, link, or path protection)

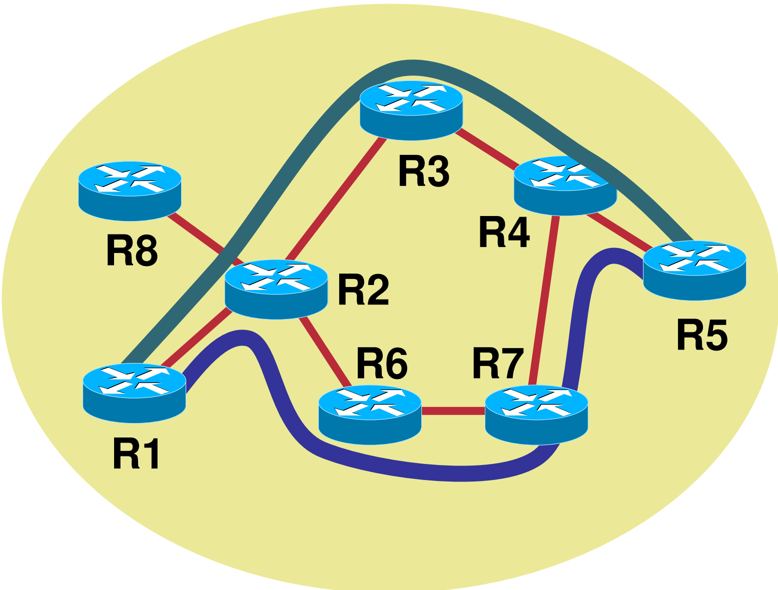
    Back up tunnels are preconfigured ➔ FRR is fast (<50 ms)

Forwarding Traffic into Tunnels



- Static
- Autoroute
- Policy-Based Routing (PBR)
- Class-Based Tunnel Selection (CBTS)

MPLS TE and QoS

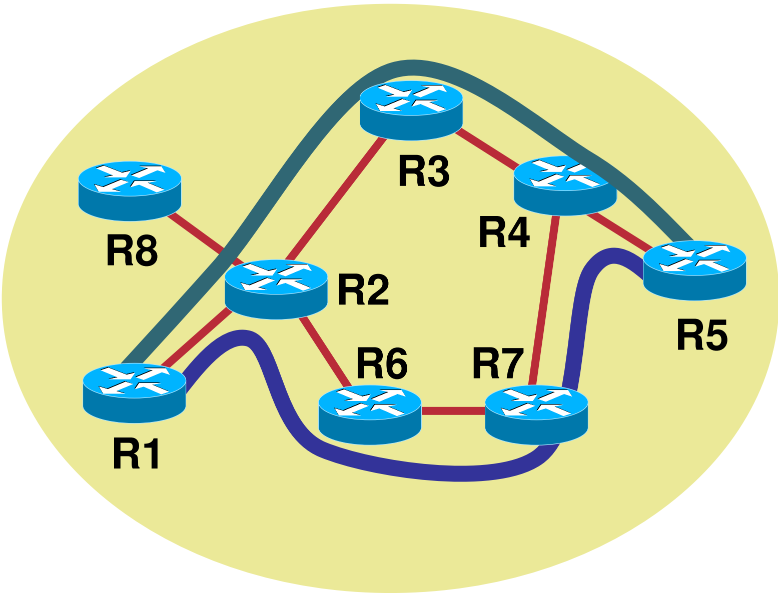


Why Is TE Important for QoS?

Separate Tunnels Can Be Made to Offer Different Latencies to the Same Destination, e.g.—

- **Different physical path lengths/hops**
- **Alignment of tunnels along paths of different degrees of traffic congestion**

DiffServ-Aware Traffic Engineering (DS-TE)



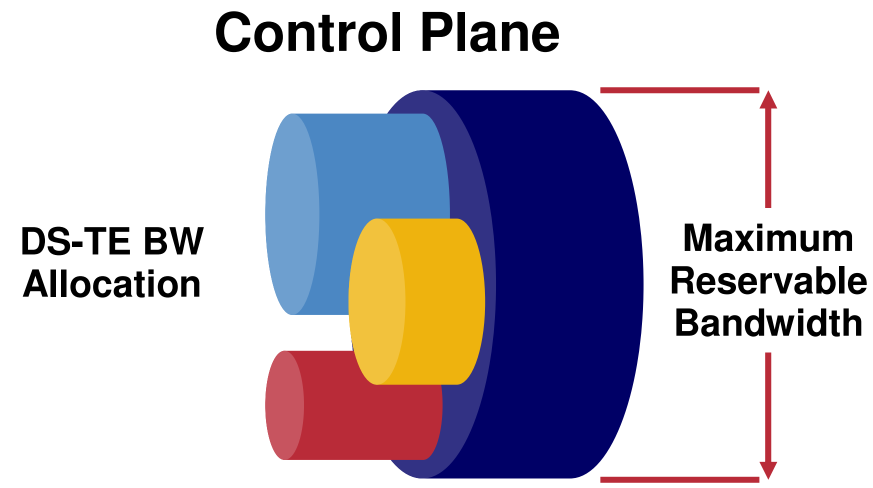
Brings Per-Class Dimension to MPLS TE

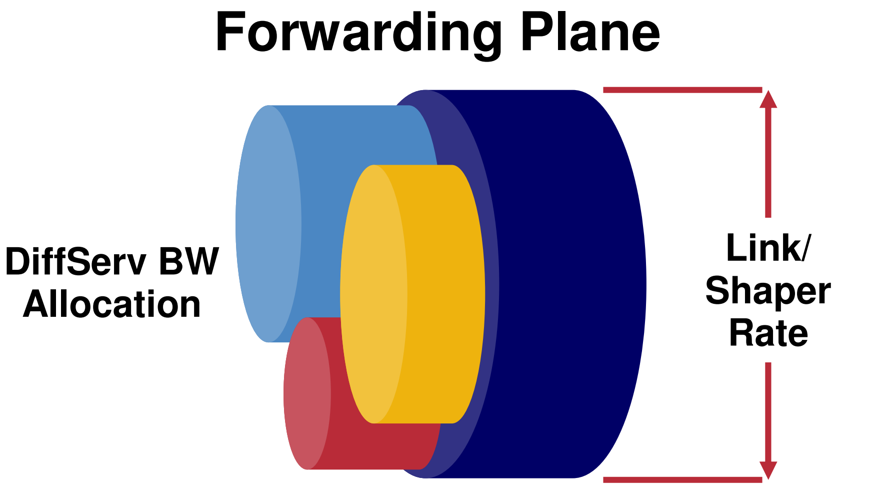
- **Per-class constrained-based routing**
- **Per-class admission control**

Low-Latency TE LSP with Reserved BW

Best-Effort TE LSP

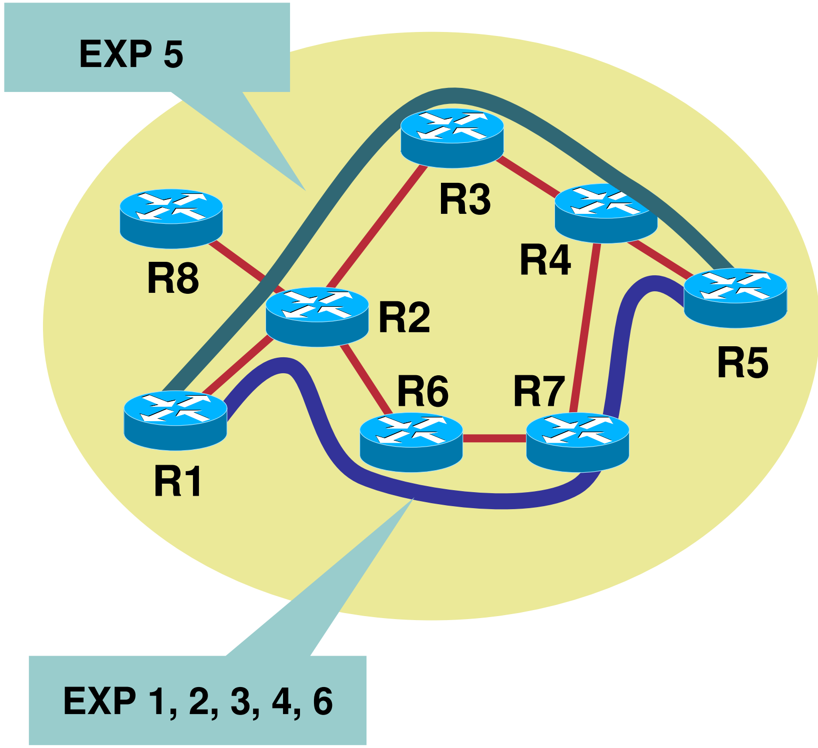
DiffServ-Aware Traffic Engineering (DS-TE)





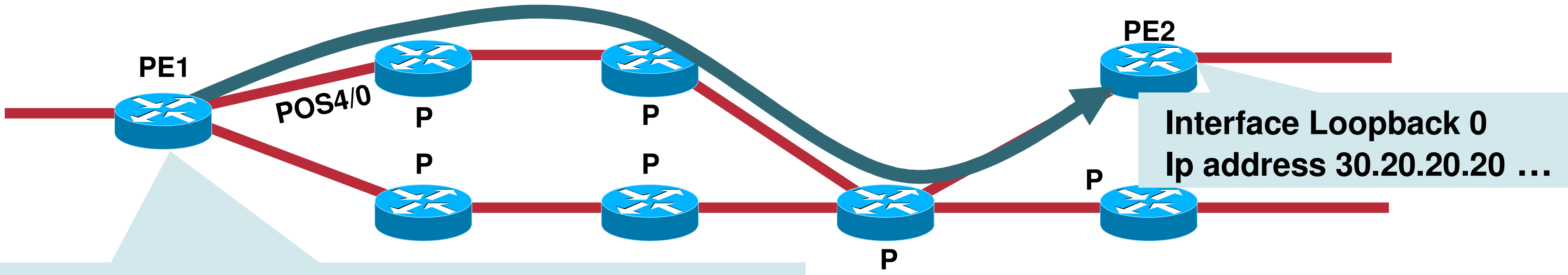
- **Link BW distributed in pools**
  Up to eight BW pools
- **Different BW pool models**
- **Unreserved BW per TE class computed using BW pools and existing reservations**
- **Unreserved BW per TE class advertised via IGP**

Class-Based Tunnel Selection: CBTS



- **EXP-based selection between multiple tunnels to same destination**
- **Tunnels configured with EXP values to carry**
- **Tunnels may be configured as default**
- **VRF aware**
- **Simplifies use of DS-TE tunnels**
- **Similar operation to ATM/FR VC bundles**

Class-Based Tunnel Selection: Config




```
Interface Loopback0
  ip address 30.10.10.10 255.255.255.255
Interface POS4/0
  Desc  PE1 to PE2
  ip address 30.1.10.1 255.255.255.252
  max-reserved-bandwidth 100
  service-policy out iPE-out-policy
  mpls traffic-eng tunnels
  tag-switching ip
  ip rsvp bandwidth 150000 sub-pool 75000
Interface Tunnel0
  Desc PE1 to PE2 tunnel (for realtime class)
  ip unnumbered Loopback0
  tunnel-destination 30.20.20.20
  tunnel mode mpls traffic-eng
  tunnel mpls traffic-eng priority 0 0
  tunnel mpls traffic-eng bandwidth sub-pool 50000
  tunnel mpls traffic-eng exp 5
  tunnel mpls traffic-eng path-option 10 dynamic
```

- **Tunnel for realtime traffic (EXP 5)**
- **No tunnel for other traffic**
- **CBTS selects EXP 5 for tunnel**
- **Other options for traffic selection:**
  - **Static routing**
  - **Policy-Based Routing (PBR)**

Summary: DiffServ, MPLS TE, DiffServ TE

- **Aggregate capacity planning**

  Adjust link capacity to expected link load

- **MPLS DiffServ**

  Adjust class capacity to expected class load

- **MPLS traffic engineering**

  Adjust link load to actual link capacity

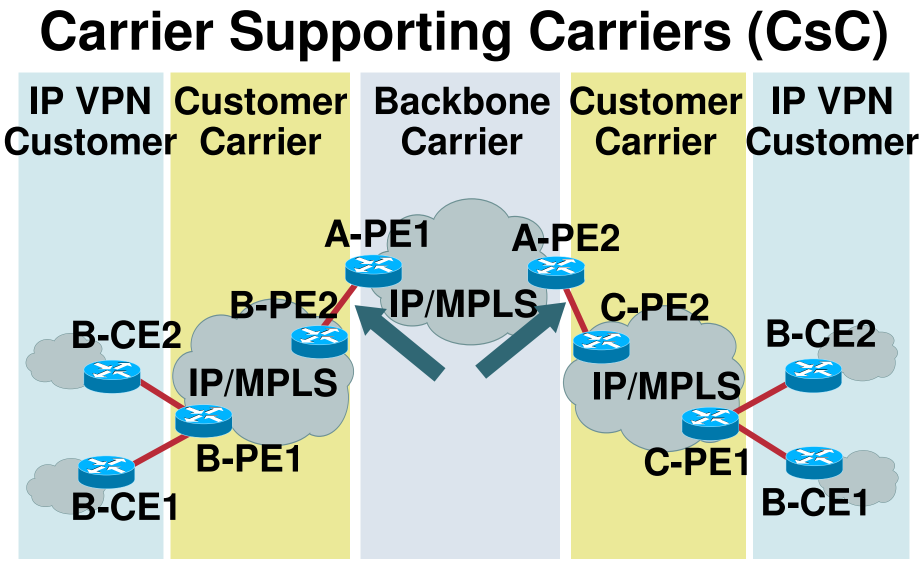
- **MPLS DiffServ-Aware TE (DS-TE)**

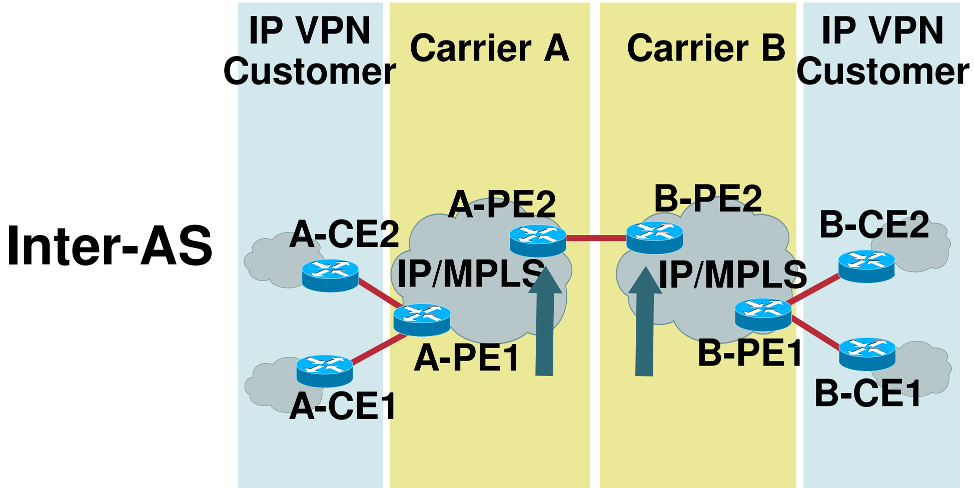
  Adjust class load to actual class capacity

InterProvider QoS

47

InterProvider QoS

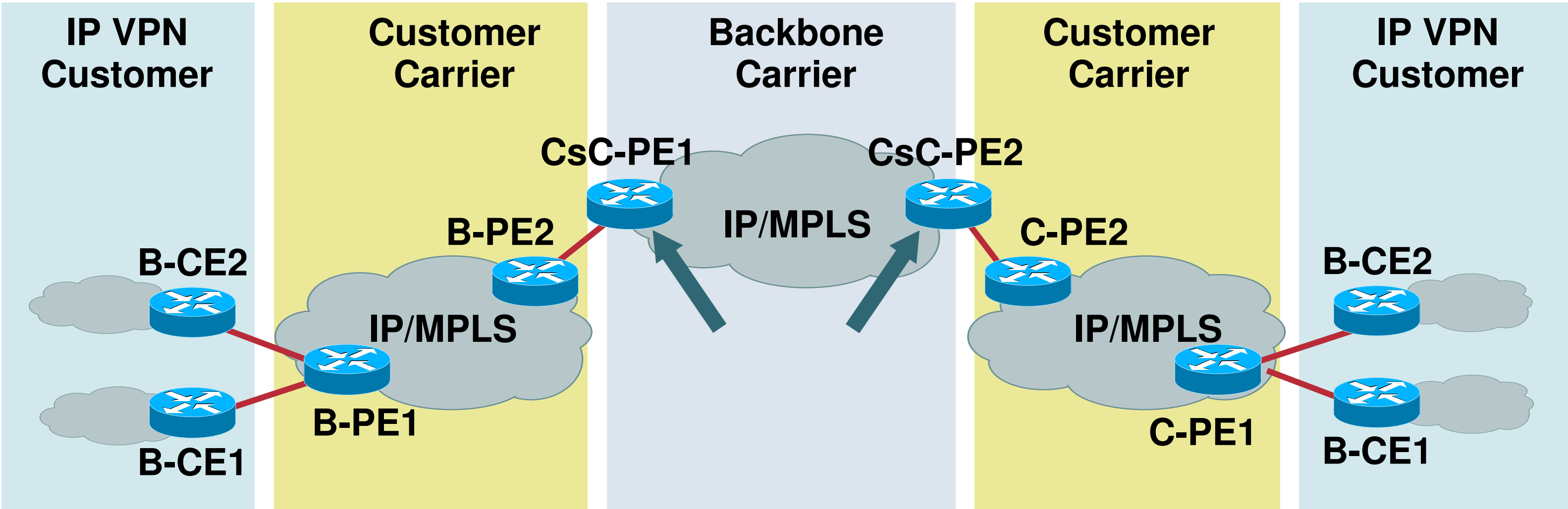




- **Needs complex coordination between providers**
  - Number of classes
  - Markings
  - SLAs and ownership
  - Consistency in SLA measurement
  - Time synchronization
- **End user may receive least common denominator**
- **MPLS DiffServ tunnel modes supports CsC hierarchies**
- **Tunnel modes may differ at different levels in a hierarchy**

InterProvider QoS: Carrier Supporting Carrier

Carrier Supporting Carriers (CsC)

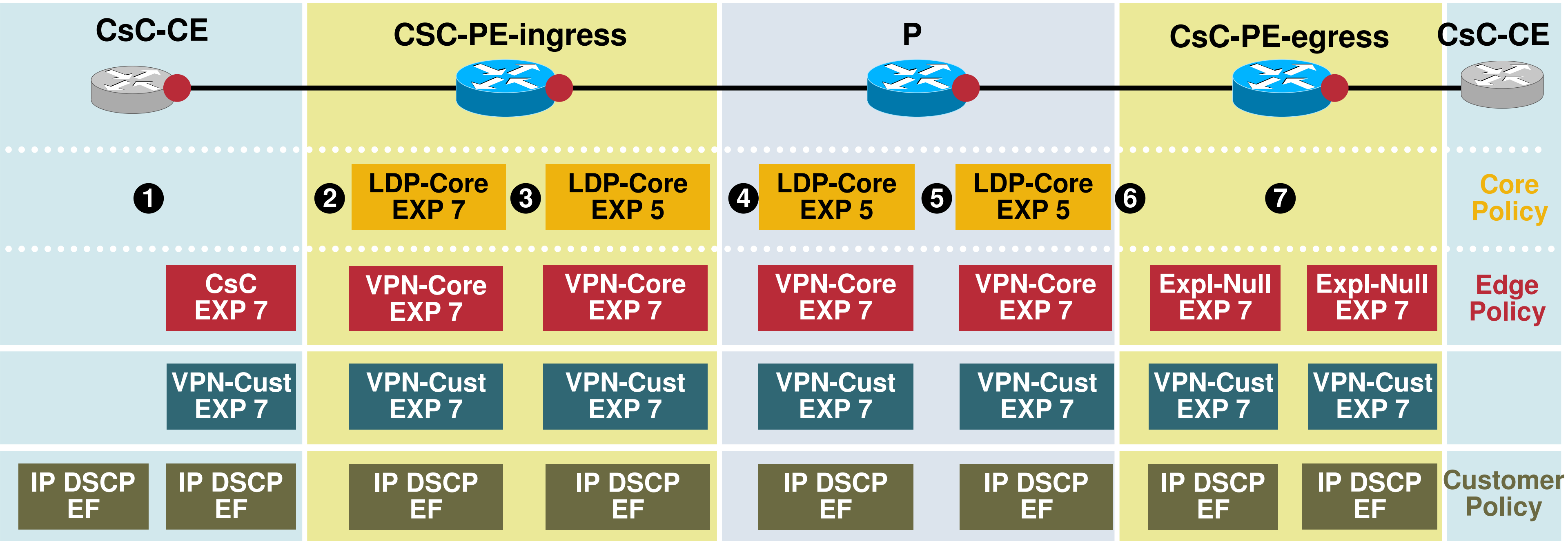


Carrier's Carrier Imposes an Additional CsC Label at Ingress CsC PE

CsC Label Dropped at Egress CsC PE

CsC Label's EXP Modified at Ingress CsC PE to Reflect QoS Policy of Backbone Carrier

Carrier Supporting Carrier: Tunnel Hierarchy



1
- Push CsC and VPN labels
- Set EXP according to Edge Policy
- PHB on Edge Policy
- Preserve DSCP Customer Policy

2
- Swap CsC to VPN-core label
- MUST preserve Edge EXP

3
- Push LDP label
- SET-TOP-MOST label to Core Policy
- PHB on Core Policy

4
- Swap LDP label
- PHB on Core Policy

5
- Use LDP explicit-Null Label
- PHB according to Core Policy

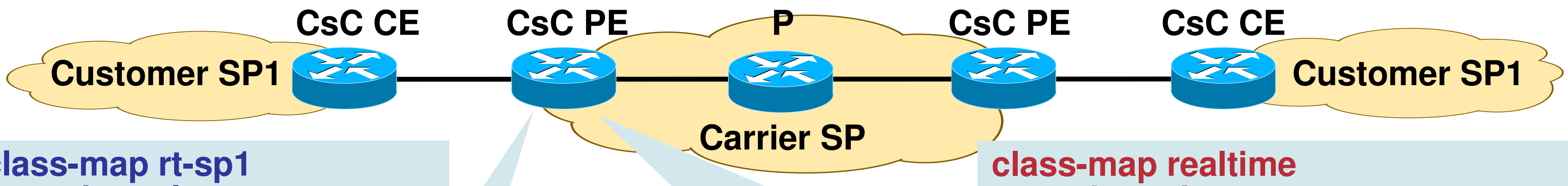
6
- Pop Core labels (LDP and VPN)
- Push Explicit Null
- Preserve Edge EXP

7
- PHB on Edge Policy

- CBWFQ (MDRR), WRED

Traffic Direction

Carrier Supporting Carrier: Tunnel Hierarchy




```
class-map rt-sp1
  match  mpls exp 7
class-map bus-lat-sp1
  match  mpls exp 4

policy-map sp2-in
class rt-sp1
   set mpls exp imposition 5
class bus-lat-sp1
   set mpls exp imposition 3
class class-default
   set mpls exp imposition 0
!
 interface POS1/0
 description to CsC CE
 service-policy input sp1-in
```

Traffic Direction

Class	SP1 EXP	CsC SP EXP
Realtime	7	5
Business	4	3
Best Effort	0	0

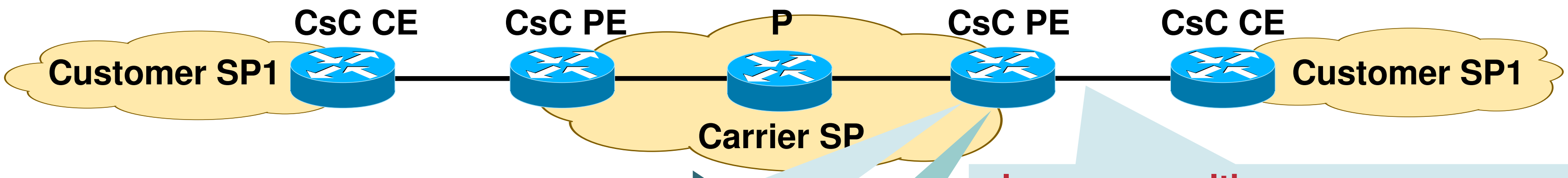
```
class-map realtime
  match  mpls exp 5
class-map bus-lat
  match mpls exp 3

policy-map sp1-out
class realtime
   priority
   queue-limit 500 packets
 class bus-lat
   bandwidth remaining percent 90
   random detect prec-based
   random-detect precedence 3 ......
 class class-default
   bandwidth remaining percent 10
   random detect prec-based
   random-detect precedence 0 ......

 interface POS4/0
 description to P router
 service-policy output sp1-out

!
```

Variations:

- Class aggregation
- Class separation

EXP Imposition at Ingress CsC PE

Carrier Supporting Carrier: Tunnel Hierarchy




```
class-map rt-sp1
  match  mpls exp 5
class-map bus-lat-sp1
  match  mpls exp 3

policy-map CsC-in
class rt-sp1
   set qos-group 5
   set discard-class 5
class bus-lat-sp1
   set qos-group 3
   set discard-class 3
class class-default
   set qos-group 0
   set discard-class 0
!
 interface POS1/0
 description to CsC CE
 service-policy input CsC-in
```

Traffic Direction

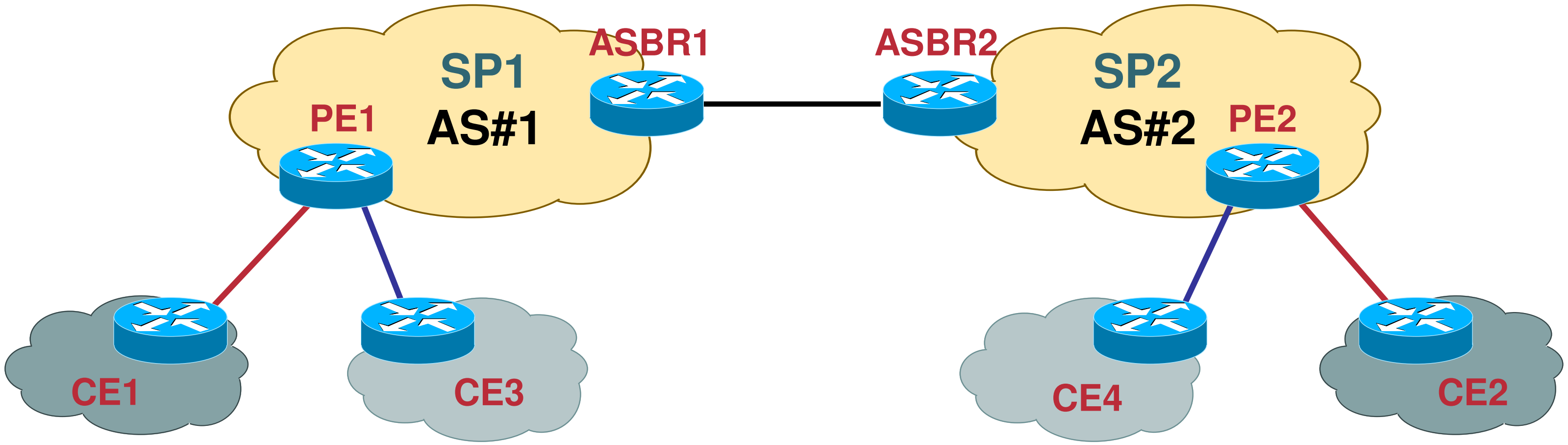
Advertises Explicit Null
"mpls ldp explicit-null"

Class	CsC SP EXP	SP1 EXP
Realtime	5	7
Business	3	4
Best Effort	0	0

```
class-map realtime
  match  qos-group 5
class-map bus-lat
  match qos-group 3

policy-map CsC-out
class realtime
   priority
   queue-limit 500 packets
 class bus-lat
   bandwidth remaining percent 90
   random detect discard-class-based
   random-detect discard-class 3 ……
 class class-default
   bandwidth remaining percent 10
   random detect discard-class-based
   random-detect discard-class 0 ……

 interface POS4/0
 description to P router
 service-policy output CsC-out

!
```

QoS at Egress CsC PE

InterProvider QoS with InterAS



One of the Two SPs Can Mark Traffic at Its ASBR to Reflect the Other SP's Policies

Class	SP1 EXP		CsC SP EXP
Realtime	7	⟷	5
Business	4	⟷	3
Best Effort	0	⟷	0

InterProvider QoS with InterAS


```
class-map rt-sp1
  match  mpls exp 7
class-map bus-lat-sp1
  match  mpls exp 4

policy-map sp1-in
class rt-sp1
   set mpls exp topmost 5
   set qos-group 5
   set discard-class 5
 class bus-lat-sp1
   set mpls exp topmost 3
   set qos-group 3
   set discard-class 3
class class-default
   set mpls exp topmost 0
   set qos-group 0
   set discard-class 0
!
 interface POS1/0
 description to ASBR2
 service-policy input sp1-in

!
```

Traffic Direction

Class	SP1 EXP	CsC SP EXP
Realtime	7	5
Business	4	3
Best Effort	0	0

EXP Translation at ABSR1

```
class-map realtime
  match  qos-group 5
class-map bus-lat
  match qos-group 3

policy-map sp1-out
class realtime
   priority
   queue-limit 500 packets
 class bus-lat
   bandwidth remaining percent 90
   random detect discard-class-based
   random-detect discard-class 3 ……
 class class-default
   bandwidth remaining percent 10
   random detect discard-class-based
   random-detect discard-class 0 ……

 interface POS4/0
 description to ASBR2
 service-policy output sp1-out

!
```

InterProvider QoS Summary

Business Model Is Evolving for InterProvider QoS

Complex Coordination Between/ Among SPs Needed to Provide SLA to Customers Today

- **SLA ownership**
- **Consistency in SLA measurement**
- **Synchronizing time**

Q and A